UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/921,015	08/01/2001	Ronald Samuel Barchi	006004.00013	5803

22909 7590 08/22/2006

BANNER & WITCOFF, LTD.
1001 G STREET, N.W.
WASHINGTON, DC 20001-4597

EXAMINER	
MOFIZ, APU M	

ART UNIT	PAPER NUMBER
2165	

DATE MAILED: 08/22/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/921,015	BARCHI ET AL.	
	Examiner	Art Unit	
	Apu M. Mofiz	2165	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *10 May 2004*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-26* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-26* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *01 August 2001* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date *05/24/2004*.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

Examiner's Response to Applicant's Remarks

1. Applicant's arguments submitted on 05/10/2004 with respect to claims 1-26 have been reconsidered but are not deemed persuasive for the reasons set forth below.

Examiner's Responses to Applicant's Remarks are listed below:

2. Applicant argues (under REMARKS section) that, Ambrosini does not teach a method for processing calls to a directory, comprising: receiving a call; evaluating the call according to one or more rules governing data that may be included in the directory; and processing the call based upon the evaluation of the call according to the one or more previously determined rules."

Examiner respectfully disagrees. Ambrosini teaches a method for processing calls (i.e. "directory servers can be defined as repositories of attributes/value pairs that clients can use to locate entries and attributes using structured queries." (Col 1, lines 29-31) ... "Each DA system typically organizes data stored therein differently from other data stored in other DA systems." (col 1, lines 40-43) ... "**LDAP** represents a simple, albeit powerful directory service which is capable of performing powerful **directory service queries as well as allowing clients to issue commands that add, delete or modify directory service entries**." (col 2, lines 20-25) ... "Furthermore, through standard interfaces, **the schema or organization of the**

attributes in an LDAP directory are obtainable." (col 2, lines 34-36) ... "**LDAP permits** a user **to control which attributes are required and allowed for a particular object class, thus determining the schema rules that the entry must obey**." (col 3, lines 15-18) ... "However it is important to note that LDAP is not an X.500 directory. Rather it is the protocol between parties **transacting business relating to any hierarchical, attribute-based directory**" (col 3, lines 19-23) ... "LDAP defines operations for interrogating and updating the directory. Furthermore, **LDAP provides operations for adding and deleting an entry from the directory, changing an existing entry, and changing the name of an entry**." (col 3, lines 25-30) ... "At the lowest level, any client requiring access to directory information connect to an LDAP server over a TCP connection. Subsequently , the client can transmit LDAP commands to the LDAP server over the TCP/IP connection." (col 4, lines 31-35) ... "LDAPv3.1 provides a "plug-in" architecture which permits a third party provider to integrate services into an LDAP server" (col 5, lines 35-37) ... "An **LDAP server plug-in is a shared object or library containing**

functions external to the functions provided with the LDAP server. Plug-in functions can be written to perform the following tasks: **Validating data before the server performs an LDAP operation on the data**" (col 5, lines 39-45) ... "Notably, the LDAP server can invoke a plug-in function at several points in the process of servicing an LDAP request. For example the LDAP server can call the LDAP server plug-in functions before executing an LDAP operation; **when the LDAP server is to add, modify, delete, or find entries in the database**" (col 5, lines 52-58) ... "Further, if the LDAP server plug-in function is invoked before an LDAP operation executes, **the plug-in function can prevent the LDAP operation from executing**. For example, **a plug-in function can validate data before new entry is added to the directory**." (col 6, lines 40-45). The preceding text excerpts clearly indicate that various Directory Assistance Systems keep their data in various LDAP directories/databases at respective LDAP servers. The data at various databases are different. **Each database schema or schema rules** define what and how the data should be e.g., format/types of data etc. LDAP clients (i.e., any application/user that conforms to the LDAP protocol including DA

systems) can communicate or send commands (or **transaction request** e.g., to search, add, delete or modify some data in the database) to the directories/databases. Before the data (i.e., attributes or attribute values for an object and schema contains attributes) can be added to the directory/database it has to conform to the schema rules. Ambrosini introduces a plug-in function that is external to the functions of the LDAP server functions. The plug-in function validates the data i.e., it verifies if the data conforms to the schema rules or any other rules that a user defines in the plug-in function before it can be applied to the directory/database. Therefore the LDAP server monitors and intercepts all calls/transactions (i.e., client commands or transaction request to perform some operation on the LDAP directory database) and invokes the plug-in function (i.e., diverts/redirects the data to the plug-in function/ rule validator) to verify the data using defined rules e.g., user defined rules or schema rules (a particular types of rule would be descriptive data and would be considered non-statutory. Therefore if the plug-in function is defined to only have add, modify or delete related validation functionality, the LDAP server need not invoke the plug-in function before sending/forwarding the data to the directory/database. Finally, **Examiner believes that all of the Applicant's arguments are explicitly addressed.**) to a directory (i.e. "In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories.") (col 2, lines 16-67), comprising: receiving a call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67); evaluating (i.e. *"determining the schema rules that the entry must obey."*) (col 3, lines 15-18; col 5, lines 35-67) the call (Fig. 2; col 2, lines 16-67) according to one or more rules (i.e. *"LDAP permits a user to control which attributes are required and allowed for a particular object class, thus determining the schema rules*

that the entry must obey." ... "Plug-in functions can be written to perform the following tasks: Validating data before the server performs an LDAP operation on the data;") (col 3, lines 15-18; col 5, lines 35-67) governing data that may be included in the directory (col 2, lines 16-67); and processing the call (Fig. 2; col 2, lines 16-67) based upon the evaluation (i.e. *"determining the schema rules that the entry must obey."*) (col 3, lines 15-18; col 5, lines 35-67) of the call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67) according to the one or more previously determined rules (i.e. *"LDAP permits a user to control which attributes are required and allowed for a particular object class, thus determining the schema rules that the entry must obey." ... "Plug-in functions can be written to perform the following tasks: Validating data before the server performs an LDAP operation on the data;"*) (col 3, lines 15-18; col 5, lines 35-67).

Applicant argues that Ambrosini does not teach a rule attribute enforcer, comprising: a rule validator for determining if attributes in a call to a directory comply with rules governing data that may be included in the directory; and a transaction monitor for intercepting calls to the directory, such that the transaction monitor diverts intercepted calls to the rule validator that include a request to add data to the directory, a request to modify data in the directory, or a request to delete data from the directory; and forwards intercepted calls to the directory that do not include a request to add data to the directory, a request to modify data in the directory, or a request to delete data from the directory.

Examiner respectfully disagrees. Ambrosini teaches a rule attribute enforcer (i.e. "directory servers can be defined as repositories of attributes/value pairs that clients can use to locate entries and attributes using structured queries." (Col 1, lines 29-31) ... "Each DA system

typically organizes data stored therein differently from other data stored in other DA systems." (col 1, lines 40-43) ... "**LDAP** represents a simple, albeit powerful directory service which is capable of performing powerful **directory service queries as well as allowing clients to issue commands that add, delete or modify directory service entries**." (col 2, lines 20-25) ... "Furthermore, through standard interfaces, **the schema or organization of the attributes in an LDAP directory** are obtainable." (col 2, lines 34-36) ... "**LDAP permits** a user **to control which attributes are required and allowed for a particular object class, thus determining the schema rules that the entry must obey**." (col 3, lines 15-18) ... "However it is important to note that LDAP is not an X.500 directory. Rather it is the protocol between parties **transacting business relating to any hierarchical, attribute-based directory**" (col 3, lines 19-23) ... "LDAP defines operations for interrogating and updating the directory. Furthermore, **LDAP provides operations for adding and deleting an entry from the directory, changing an existing entry, and changing the**

name of an entry." (col 3, lines 25-30) ... "At the lowest level, any client requiring access to directory information connect to an LDAP server over a TCP connection. Subsequently , the client can transmit LDAP commands to the LDAP server over the TCP/IP connection." (col 4, lines 31-35) ... "LDAPv3.1 provides a "plug-in" architecture which permits a third party provider to integrate services into an LDAP server" (col 5, lines 35-37) ... "An **LDAP server plug-in is a shared object or library containing functions external to the functions provided with the LDAP server**. Plug-in functions can be written to perform the following tasks: **Validating data before the server performs an LDAP operation on the data**" (col 5, lines 39-45) ... "Notably, the LDAP server can invoke a plug-in function at several points in the process of servicing an LDAP request. For example the LDAP server can call the LDAP server plug-in functions before executing an LDAP operation; **when the LDAP server is to add, modify, delete, or find entries in the database**" (col 5, lines 52-58) ... "Further, if the LDAP server plug-in function is invoked before an LDAP operation executes, **the plug-in function can prevent the LDAP operation from executing**. For example, **a plug-in function**

can validate data before new entry is added to the directory."

(col 6, lines 40-45). The preceding text excerpts clearly indicate that various Directory Assistance Systems keep their data in various LDAP directories/databases at respective LDAP servers. The data at various databases are different. **Each database schema or schema rules** define what and how the data should be e.g., format/types of data etc. LDAP clients (i.e., any application/user that conforms to the LDAP protocol including DA systems) can communicate or send commands (or **transaction request** e.g., to search, add, delete or modify some data in the database) to the directories/databases. Before the data (i.e., attributes or attribute values for an object and schema contains attributes) can be added to the directory/database it has to conform to the schema rules. Ambrosini introduces a plug-in function that is external to the functions of the LDAP server functions. The plug-in function validates the data i.e., it verifies if the data conforms to the schema rules or any other rules that a user defines in the plug-in function before it can be applied to the directory/database. Therefore the LDAP server monitors and intercepts all calls/transactions (i.e., client commands or transaction request to perform some operation on the LDAP directory database) and invokes the plug-in function (i.e., diverts/redirects the data to the plug-in function/ rule validator) to verify the data using defined rules e.g., user defined rules or schema rules (a particular types of rule would be descriptive data and would be considered non-statutory. Therefore if the plug-in function is defined to only have add, modify or delete related validation functionality, the LDAP server need not invoke the plug-in function before sending/forwarding the data to the

directory/database. Finally, **Examiner believes that all of the Applicant's arguments are explicitly addressed.**), comprising: a rule validator (i.e. plug-in function) (col 5, lines 35-67; col 6, lines 35-47) for determining (i.e. determining if the request data/ entry is valid by some validity standards/rules) (col 3, lines 15-18; col 5, lines 35-67) if attributes (i.e. *"In performing a directory query, LDAP clients can choose filter attributes in the directory tree, for example a search location, and filter the search therefrom. Sample "base" values can include "st=FL . . . c=us" or "I=Boca Raton, I=Highland Beach Directory, st=FL, . . . , c=us"."*) (col 3, lines 48-53; col 4, lines 50-67; col 5, lines 1-67) in a call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67) to a directory (i.e. *"In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories."*) (col 2, lines 16-67) comply (i.e. determining if the request data/ entry is valid by some validity standards/rules) (col 3, lines 15-18; col 5, lines 35-67) with rules (i.e. validity criteria used by the plug-in function) (col 5, lines 35-67; col 6, lines 35-47) governing data that may be included in the directory (i.e. *"In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories."*) (col 2, lines 16-67); and a transaction (i.e. add, modify or delete transaction) (col 5, lines 35-67; col 6, lines 35-47) monitor (i.e. the LDAP server) (col 5, lines 50-67) for intercepting (i.e. *"An LDAP client can connect to an LDAP servers and transmit a request for data."*) (col 4, lines 15-30) calls (Fig. 2; col 2, lines 16-67) to the directory (col 2, lines 16-67), such that the transaction monitor (i.e. the LDAP server) (col 5, lines 50-67) diverts (i.e. *"In most cases, when the LDAP server calls an LDAP server plug-in function, the LDAP server passes a parameter block to the plug-in function."*) (col 5, lines 35-67; col 6, lines 35-47) intercepted calls to the rule validator (i.e. plug-in function) (col 5, lines 35-67; col 6, lines 35-47) that include a request to add data (i.e. *"LDAP defines operations for interrogating and updating the directory. Furthermore, LDAP provides operations for adding and deleting an entry from the directory, changing an existing entry, and changing the name*

of an entry. Still the primary operation of LDAP is to search for information stored in the directory." ... "Further, if the LDAP server plug-in function is invoked before an LDAP operation executes, the plug-in function can prevent the LDAP operation from executing. For example, a plug-in function can validate data before a new entry is added to the directory.") (col 3, lines 25-30; col 6, lines 35-47) to the directory (col 2, lines 16-67), a request to modify data (col 3, lines 25-30; col 6, lines 35-47) in the directory (col 2, lines 16-67), or a request to delete data (col 3, lines 25-30; col 6, lines 35-47) from the directory (col 2, lines 16-67); and forwards (i.e. if the client request does not involve adding, deleting or modifying data then the request is for searching data and the searching request does not need validation and is merely forwarded/ sent to the directory) (col 3, lines 25-30; col 6, lines 35-47) intercepted calls (Fig. 2; col 2, lines 16-67 to the directory (col 2, lines 16-67) that do not include a request to add data (col 3, lines 25-30; col 6, lines 35-47) to the directory (col 2, lines 16-67), a request to modify data (col 3, lines 25-30; col 6, lines 35-47) in the directory (col 2, lines 16-67), or a request to delete data (col 3, lines 25-30; col 6, lines 35-47) from the directory (col 2, lines 16-67).

Any other arguments by the applicant are more limiting than the claimed language.

Claim Rejections - 35 USC § 102

3. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the

applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

4. Claims 1-26 are rejected under 35 U.S.C. 102(e) as being anticipated by Ambrosini et al. (U.S. Patent No. 6,609,121 and Ambrosini hereinafter).

As to claim 1, Ambrosini teaches a method for processing calls (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67) to a directory (i.e. "In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories.") (col 2, lines 16-67), comprising: receiving a call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67); evaluating (i.e. *"determining the schema rules that the entry must obey."*) (col 3, lines 15-18; col 5, lines 35-67) the call (Fig. 2; col 2, lines 16-67) according to one or more rules (i.e. *"LDAP permits a user to control which attributes are required and allowed for a particular object class, thus determining the schema rules that the entry must obey." ... "Plug-in functions can be written to perform the following tasks: Validating data before the server performs an LDAP operation on the data;"*) (col 3, lines 15-18; col 5, lines 35-67) governing data that may be included in the directory (col 2, lines 16-67); and processing the call (Fig. 2; col 2, lines 16-67) based upon the evaluation (i.e. *"determining the schema rules that the entry must obey."*) (col 3, lines 15-18; col 5, lines 35-67) of the call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67) according to the one or more previously determined rules (i.e. *"LDAP permits a user to control which attributes are required and allowed for a particular object class, thus determining the schema rules that the entry must obey." ... "Plug-in functions can be written to perform the following tasks: Validating data before the server performs an LDAP operation on the data;"*) (col 3, lines 15-18; col 5, lines 35-67).

As to claim 2, Ambrosini teaches that the step of evaluating (i.e. *"determining the schema rules that the entry must obey."*) (col 3, lines 15-18; col 5, lines 35-67) the call includes determining if the call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67) includes a request to add data (i.e. *"LDAP represents a simple, albeit powerful directory service which is capable of performing powerful directory service queries as well as allowing clients to issue commands that add, delete or modify directory service entries."*) (col 2, lines 16-67) to the directory (col 2, lines 16-67), a request to modify data (col 2, lines 16-67) in the directory, or a request to delete (col 2, lines 16-67) data from the directory (i.e. *"In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories."*) (col 2, lines 16-67).

As to claim 3, Ambrosini teaches wherein if the call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67) does not include a request to add data (i.e. *"LDAP defines operations for interrogating and updating the directory. Furthermore, LDAP provides operations for adding and deleting an entry from the directory, changing an existing entry, and changing the name of an entry. Still the primary operation of LDAP is to search for information stored in the directory."* ... *"Further, if the LDAP server plug-in function is invoked before an LDAP operation executes, the plug-in function can prevent the LDAP operation from executing. For example, a plug-in function can validate data before a new entry is added to the directory."*) (col 3, lines 25-30; col 6, lines 35-47) to the directory (i.e. *"In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories."*) (col 2, lines 16-67), a request to modify data (col 3, lines 25-30; col 6, lines 35-47) in the directory, or a request to delete data (col 3, lines 25-30; col 6, lines 35-47) from the directory (col 2, lines 16-67), then the processing step includes forwarding (i.e. if the client request does not involve adding, deleting or modifying data then the request is for

searching data and the searching request does not need validation and is merely forwarded/ sent to the directory) (col 3, lines 25-30; col 6, lines 35-47) the call to the directory (col 2, lines 16-67).

As to claim 4, Ambrosini teaches wherein the call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67) is forwarded to the directory (i.e. *"In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories."*) (col 2, lines 16-67) through a directory access server (i.e. the LDAP server) (col 5, lines 50-67) controlling access (i.e. *"For example, in order to restrict searches of a directory to entries exclusively including access control lists, the search phrase "objectclass=acl" can be specified so that only entries purporting to be access control lists are located."*) (col 3, lines 10-17) to the directory (col 2, lines 16-67).

As to claim 5, Ambrosini teaches that wherein the evaluation (i.e. *"determining the schema rules that the entry must obey."*) (col 3, lines 15-18; col 5, lines 35-67) step further includes determining (i.e. determining if the request data/ entry is valid by some validity standards/rules) (col 3, lines 15-18; col 5, lines 35-67) if one or more attributes (i.e. *" In performing a directory query, LDAP clients can choose filter attributes in the directory tree, for example a search location, and filter the search therefrom. Sample "base" values can include "st=FL . . . c=us" or "I=Boca Raton, I=Highland Beach Directory, st=FL, . . . , c=us"."*) (col 3, lines 48-53; col 4, lines 50-67; col 5, lines 1-67) included in the call (Fig. 2; col 2, lines 16-67) comply (i.e. determining if the request data/ entry is valid by some validity standards/rules) (col 3, lines 15-18; col 5, lines 35-67) with one or more rules in a set of rules (i.e. the validity standards/rules or schema rules; a set can consist of one or more entities in it.) (col 3, lines 15-18; col 5, lines 35-67) when the call (Fig. 2; col 2, lines 16-67) includes a request (Fig. 2; col 2, lines 16-67) to add data (i.e. *"LDAP defines operations for interrogating and updating the directory. Furthermore, LDAP provides operations for adding and*

deleting an entry from the directory, changing an existing entry, and changing the name of an entry. Still the primary operation of LDAP is to search for information stored in the directory." ... "Further, if the LDAP server plug-in function is invoked before an LDAP operation executes, the plug-in function can prevent the LDAP operation from executing. For example, a plug-in function can validate data before a new entry is added to the directory.") (col 3, lines 25-30; col 6, lines 35-47) to the directory (i.e. *"In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories.")* (col 2, lines 16-67), a request to modify data (col 3, lines 25-30; col 6, lines 35-47) in the directory (col 2, lines 16-67), or a request to delete data (col 3, lines 25-30; col 6, lines 35-47) from the directory (col 2, lines 16-67).

As to claim 6, Ambrosini teaches that wherein the processing step includes forwarding (i.e. to send the client request to the directory if the validity of the client request is fulfilled) (col 3, lines 25-30; col 6, lines 35-47) the call (Fig. 2; col 2, lines 16-67) to the directory (col 2, lines 16-67) when the one or more attributes (i.e. *" In performing a directory query, LDAP clients can choose filter attributes in the directory tree, for example a search location, and filter the search therefrom. Sample "base" values can include "st=FL . . . c=us" or "I=Boca Raton, I=Highland Beach Directory, st=FL, . . . , c=us".")* (col 3, lines 48-53; col 4, lines 50-67; col 5, lines 1-67) included in the call (Fig. 2; col 2, lines 16-67) comply (i.e. determining if the request data/ entry is valid by some validity standards/rules) (col 3, lines 15-18; col 5, lines 35-67) with each of the one or more rules in the first set of rules (i.e. the validity standards/rules or schema rules; a set can consist of one or more entities in it.) (col 3, lines 15-18; col 5, lines 35-67).

As to claim 7, Ambrosini teaches wherein the call (Fig. 2; col 2, lines 16-67) is forwarded (i.e. to send the client request to the directory if the validity of the client request is fulfilled) (col 3, lines 25-30; col 6, lines 35-47) to the directory (i.e. *"In LDAP, the basic unit of information consists of an*

entry. Entries are stored in a directories.") (col 2, lines 16-67) through a directory access server (i.e. the LDAP server) (col 5, lines 50-67) controlling access (i.e. *"For example, in order to restrict searches of a directory to entries exclusively including access control lists, the search phrase "objectclass=acl" can be specified so that only entries purporting to be access control lists are located."*) (col 3, lines 10-17) to the directory (col 2, lines 16-67).

As to claim 8, Ambrosini teaches that wherein the processing step includes forwarding (i.e. to send/ return an error message to the client) (col 6, lines 35-60) an error message (i.e. "For example, a plug-in function can validate data before a new entry is added to the directory. If the data is invalid, the plug-in function can abort the LDAP add operation and return an error message to the LDAP client) (col 6, lines 35-48) to a source (i.e. the LDAP client) (col 6, lines 35-60) of the call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67) when the one or more attributes (i.e. *"In performing a directory query, LDAP clients can choose filter attributes in the directory tree, for example a search location, and filter the search therefrom. Sample "base" values can include "st=FL . . . c=us" or "I=Boca Raton, I=Highland Beach Directory, st=FL, . . . , c=us"."*) (col 3, lines 48-53; col 4, lines 50-67; col 5, lines 1-67) included in the call (Fig. 2; col 2, lines 16-67) do not comply (i.e. determining if the request data/ entry is valid by some validity standards/rules) (col 3, lines 15-18; col 5, lines 35-67) with each of the one or more rules in the set of rules (i.e. the validity standards/rules or schema rules; a set can consist of one or more entities in it.) (col 3, lines 15-18; col 5, lines 35-67).

As to claim 9, Ambrosini teaches that wherein the processing step includes forwarding (i.e. to send the client request to the directory if the validity of the client request is fulfilled) (col 3, lines 25-30; col 6, lines 35-47) the call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines

16-67) to the directory (i.e. *"In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories."*) (col 2, lines 16-67) when the one or more attributes (i.e. *"In performing a directory query, LDAP clients can choose filter attributes in the directory tree, for example a search location, and filter the search therefrom. Sample "base" values can include "st=FL . . . c=us" or "I=Boca Raton, I=Highland Beach Directory, st=FL, . . . , c=us"."*) (col 3, lines 48-53; col 4, lines 50-67; col 5, lines 1-67) included in the call comply (i.e. determining if the request data/ entry is valid by some validity standards/rules) (col 3, lines 15-18; col 5, lines 35-67) with at least one of the one or more rules in the set of rules (i.e. the validity standards/rules or schema rules; a set can consist of one or more entities in it.) (col 3, lines 15-18; col 5, lines 35-67).

As to claim 10, Ambrosini teaches that wherein the call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67) is forwarded to the directory (i.e. *"In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories."*) (col 2, lines 16-67) through a directory access server (i.e. the LDAP server) (col 5, lines 50-67) controlling access (i.e. *"For example, in order to restrict searches of a directory to entries exclusively including access control lists, the search phrase "objectclass=acl" can be specified so that only entries purporting to be access control lists are located."*) (col 3, lines 10-17) to the directory (col 2, lines 16-67).

As to claim 11, Ambrosini teaches wherein the processing step includes forwarding (i.e. to send/ return an error message to the client) (col 6, lines 35-60) an error message (i.e. *"For example, a plug-in function can validate data before a new entry is added to the directory. If the data is invalid, the plug-in function can abort the LDAP add operation and return an error message to the LDAP client"*) (col 6, lines 35-48) to a source (i.e. the LDAP client) (col 6, lines 35-60) of the call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67) when the one or more attributes (i.e. *"In*

performing a directory query, LDAP clients can choose filter attributes in the directory tree, for example a search location, and filter the search therefrom. Sample "base" values can include "st=FL . . . c=us" or "I=Boca Raton, I=Highland Beach Directory, st=FL, . . . , c=us".") (col 3, lines 48-53; col 4, lines 50-67; col 5, lines 1-67) included in the call (Fig. 2; col 2, lines 16-67) do not comply (i.e. determining if the request data/ entry is valid by some validity standards/rules) (col 3, lines 15-18; col 5, lines 35-67) with any of the one or more rules in the set of rules (i.e. the validity standards/rules or schema rules; a set can consist of one or more entities in it.) (col 3, lines 15-18; col 5, lines 35-67).

As to claim 12, Ambrosini teaches that wherein the evaluation (i.e. *"determining the schema rules that the entry must obey."*) (col 3, lines 15-18; col 5, lines 35-67) step further includes determining (i.e. determining if the request data/ entry is valid by some validity standards/rules) (col 3, lines 15-18; col 5, lines 35-67) if one or more attributes (i.e. *"In performing a directory query, LDAP clients can choose filter attributes in the directory tree, for example a search location, and filter the search therefrom. Sample "base" values can include "st=FL . . . c=us" or "I=Boca Raton, I=Highland Beach Directory, st=FL, . . . , c=us"*) (col 3, lines 48-53; col 4, lines 50-67; col 5, lines 1-67) included in the call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67) comply (i.e. determining if the request data/ entry is valid by some validity standards/rules) (col 3, lines 15-18; col 5, lines 35-67) with one or more rules in a first (i.e. the validity rule corresponding to the LDAP client's add operation that the plug-in function uses. A set can consist of one or more entities) (col 3, lines 15-18; col 5, lines 35-67) set of rules (i.e. the validity standards/rules or schema rules; a set can consist of one or more entities in it.) (col 3, lines 15-18; col 5, lines 35-67) when the call includes a request to add data (i.e. *"LDAP defines operations for interrogating and updating the directory. Furthermore, LDAP provides operations for adding and deleting an entry from the directory, changing an existing entry, and changing the name of an entry. Still the primary operation of LDAP is to search for information stored in the directory." ... "Further, if the LDAP server plug-in function is invoked before an LDAP operation*

executes, the plug-in function can prevent the LDAP operation from executing. For example, a plug-in function can validate data before a new entry is added to the directory.") (col 3, lines 25-30; col 6, lines 35-47) to the directory, determining if one or more attributes (col 3, lines 48-53; col 4, lines 50-67; col 5, lines 1-67) included in the call (Fig. 2; col 2, lines 16-67) comply with one or more rules in a second (i.e. the validity rule corresponding to the LDAP client's modify operation that the plug-in function uses. A set can consist of one or more entities) (col 3, lines 15-18; col 5, lines 35-67) set of rules (i.e. the validity standards/rules or schema rules; a set can consist of one or more entities in it.) (col 3, lines 15-18; col 5, lines 35-67) when the call (Fig. 2; col 2, lines 16-67) includes a request to modify data (col 3, lines 25-30; col 6, lines 35-47) in the directory, and determining (col 3, lines 15-18; col 5, lines 35-67) if one or more attributes (col 3, lines 48-53; col 4, lines 50-67; col 5, lines 1-67) included in the call (Fig. 2; col 2, lines 16-67) comply with one or more rules in a third (i.e. the validity rule corresponding to the LDAP client's delete operation that the plug-in function uses. A set can consist of one or more entities) (col 3, lines 15-18; col 5, lines 35-67) set of rules (i.e. the validity standards/rules or schema rules; a set can consist of one or more entities in it.) (col 3, lines 15-18; col 5, lines 35-67) when the call (Fig. 2; col 2, lines 16-67) includes a request to delete data (col 3, lines 25-30; col 6, lines 35-47) from the directory (i.e. *"In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories."*) (col 2, lines 16-67).

As to claim 13, Ambrosini teaches wherein the directory (i.e. *"In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories."*) (col 2, lines 16-67) employs the lightweight directory access protocol (i.e. LDAP) (col 2, lines 16-67).

As to claim 14, Ambrosini teaches a rule attribute enforcer (i.e. plug-in function) (col 5, lines 35-67; col 6, lines 35-47), comprising: a rule validator (i.e. plug-in function) (col 5, lines 35-67; col 6,

lines 35-47) for determining (i.e. determining if the request data/ entry is valid by some validity standards/rules) (col 3, lines 15-18; col 5, lines 35-67) if attributes (i.e. *"In performing a directory query, LDAP clients can choose filter attributes in the directory tree, for example a search location, and filter the search therefrom. Sample "base" values can include "st=FL . . . c=us" or "I=Boca Raton, I=Highland Beach Directory, st=FL, . . . , c=us". "*) (col 3, lines 48-53; col 4, lines 50-67; col 5, lines 1-67) in a call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67) to a directory (i.e. *"In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories. "*) (col 2, lines 16-67) comply (i.e. determining if the request data/ entry is valid by some validity standards/rules) (col 3, lines 15-18; col 5, lines 35-67) with rules (i.e. validity criteria used by the plug-in function) (col 5, lines 35-67; col 6, lines 35-47) governing data that may be included in the directory (i.e. *"In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories. "*) (col 2, lines 16-67); and a transaction (i.e. add, modify or delete transaction) (col 5, lines 35-67; col 6, lines 35-47) monitor (i.e. the LDAP server) (col 5, lines 50-67) for intercepting (i.e. *"An LDAP client can connect to an LDAP servers and transmit a request for data. "*) (col 4, lines 15-30) calls (Fig. 2; col 2, lines 16-67) to the directory (col 2, lines 16-67), such that the transaction monitor (i.e. the LDAP server) (col 5, lines 50-67) diverts (i.e. *"In most cases, when the LDAP server calls an LDAP server plug-in function, the LDAP server passes a parameter block to the plug-in function. "*) (col 5, lines 35-67; col 6, lines 35-47) intercepted calls to the rule validator (i.e. plug-in function) (col 5, lines 35-67; col 6, lines 35-47) that include a request to add data (i.e. *"LDAP defines operations for interrogating and updating the directory. Furthermore, LDAP provides operations for adding and deleting an entry from the directory, changing an existing entry, and changing the name of an entry. Still the primary operation of LDAP is to search for information stored in the directory." ... "Further, if the LDAP server plug-in function is invoked before an LDAP operation executes, the plug-in function can prevent the LDAP operation from executing. For example, a plug-in function can validate data before a new entry is added to the directory. "*) (col 3, lines 25-30; col 6, lines 35-47) to the

directory (col 2, lines 16-67), a request to modify data (col 3, lines 25-30; col 6, lines 35-47) in the directory (col 2, lines 16-67), or a request to delete data (col 3, lines 25-30; col 6, lines 35-47) from the directory (col 2, lines 16-67); and forwards (i.e. if the client request does not involve adding, deleting or modifying data then the request is for searching data and the searching request does not need validation and is merely forwarded/ sent to the directory) (col 3, lines 25-30; col 6, lines 35-47) intercepted calls (Fig. 2; col 2, lines 16-67 to the directory (col 2, lines 16-67) that do not include a request to add data (col 3, lines 25-30; col 6, lines 35-47) to the directory (col 2, lines 16-67), a request to modify data (col 3, lines 25-30; col 6, lines 35-47) in the directory (col 2, lines 16-67), or a request to delete data (col 3, lines 25-30; col 6, lines 35-47) from the directory (col 2, lines 16-67).

As to claim 15, Ambrosini teaches wherein when the rule validator (i.e. plug-in function) (col 5, lines 35-67; col 6, lines 35-47) determines (i.e. determining if the request data/ entry is valid by some validity standards/rules) (col 3, lines 15-18; col 5, lines 35-67) that one or more attributes (i.e. *"In performing a directory query, LDAP clients can choose filter attributes in the directory tree, for example a search location, and filter the search therefrom. Sample "base" values can include "st=FL . . . c=us" or "I=Boca Raton, I=Highland Beach Directory, st=FL, . . . , c=us"."*) (col 3, lines 48-53; col 4, lines 50-67; col 5, lines 1-67) included in a call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67) comply (i.e. determining if the request data/ entry is valid by some validity standards/rules) (col 3, lines 15-18; col 5, lines 35-67) with each of one or more rules in a set of rules (i.e. the validity standards/rules or schema rules; a set can consist of one or more entities in it.) (col 3, lines 15-18; col 5, lines 35-67), the rule validator (i.e. plug-in function) (col 5, lines 35-67; col 6, lines 35-47) forwards (i.e. to send the client request to the directory if the validity of the client request is fulfilled) (col 3, lines 25-30; col 6, lines 35-47) the call (Fig. 2; col 2, lines 16-67).

As to claim 16, Ambrosini teaches wherein the rule validator (i.e. plug-in function) (col 5, lines 35-67; col 6, lines 35-47) forwards (i.e. The LDAP server calls the plug-in function before executing the LDAP operation (e.g. add, delete, modify or search); The plug-in function validates the operation and then sends the validation result to the LDAP server and the server executes the operation on the directory) (col 5, lines 35-67; col 6, lines 35-47) the call to the transaction monitor (i.e. the LDAP server) (col 5, lines 50-67), and the transaction monitor (i.e. the LDAP server) (col 5, lines 50-67) relays (col 5, lines 35-67; col 6, lines 35-47) the call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67) to the directory (i.e. *"In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories."*) (col 2, lines 16-67).

As to claim 17, Ambrosini teaches wherein transaction monitor (i.e. the LDAP server (16), a software entity which is situated in the LDAP server) (Fig. 3; col 5, lines 50-67) relays the call to the directory (i.e. *"In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories."*) (col 2, lines 16-67) through a directory access server (i.e. the LDAP server) (col 5, lines 50-67) that controls access (i.e. *"For example, in order to restrict searches of a directory to entries exclusively including access control lists, the search phrase "objectclass=acl" can be specified so that only entries purporting to be access control lists are located."*) (col 3, lines 10-17) to the directory (col 2, lines 16-67).

As to claim 18, Ambrosini teaches wherein the rule validator (i.e. plug-in function) (col 5, lines 35-67; col 6, lines 35-47) forwards (i.e. to send the client request to the directory if the validity of the client request is fulfilled) (col 3, lines 25-30; col 6, lines 35-47) the call (i.e. client's query/ request into the

directory database) (Fig. 2; col 2, lines 16-67) to the directory (i.e. *"In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories."*) (col 2, lines 16-67).

As to claim 19, Ambrosini teaches wherein the rule validator (i.e. plug-in function) (col 5, lines 35-67; col 6, lines 35-47) forwards (i.e. The LDAP server calls the plug-in function before executing the LDAP operation (e.g. add, delete, modify or search); The plug-in function validates the operation and then sends the validation result to the LDAP server and the server executes the operation on the directory) (col 5, lines 35-67; col 6, lines 35-47) the call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67) to the directory (i.e. *"In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories."*) (col 2, lines 16-67) through a directory access server (i.e. the LDAP server) (col 5, lines 50-67) that controls access (i.e. *"For example, in order to restrict searches of a directory to entries exclusively including access control lists, the search phrase "objectclass=acl" can be specified so that only entries purporting to be access control lists are located."*) (col 3, lines 10-17) to the directory (col 2, lines 16-67).

As to claim 20, Ambrosini teaches wherein when the rule validator (i.e. plug-in function) (col 5, lines 35-67; col 6, lines 35-47) determines (i.e. determining if the request data/ entry is valid by some validity standards/rules) (col 3, lines 15-18; col 5, lines 35-67) that one or more attributes (i.e. *"In performing a directory query, LDAP clients can choose filter attributes in the directory tree, for example a search location, and filter the search therefrom. Sample "base" values can include "st=FL . . . c=us" or "I=Boca Raton, I=Highland Beach Directory, st=FL, . . . , c=us"."*) (col 3, lines 48-53; col 4, lines 50-67; col 5, lines 1-67) included in a call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67) comply (i.e. determining if the request data/ entry is valid by some validity standards/rules) (col 3, lines 15-18; col 5, lines 35-67) with at least one of one or more rules in a set of rules (i.e. the validity standards/rules or

schema rules; a set can consist of one or more entities in it.) (col 3, lines 15-18; col 5, lines 35-67), the rule validator (col 5, lines 35-67; col 6, lines 35-47) forwards (i.e. The LDAP server calls the plug-in function before executing the LDAP operation (e.g. add, delete, modify or search); The plug-in function validates the operation and then sends the validation result to the LDAP server and the server executes the operation on the directory) (col 5, lines 35-67; col 6, lines 35-47) the call (Fig. 2; col 2, lines 16-67).

As to claim 21, Ambrosini teaches wherein the rule validator (i.e. plug-in function) (col 5, lines 35-67; col 6, lines 35-47) forwards (i.e. The LDAP server calls the plug-in function before executing the LDAP operation (e.g. add, delete, modify or search); The plug-in function validates the operation and then sends the validation result to the LDAP server and the server executes the operation on the directory) (col 5, lines 35-67; col 6, lines 35-47) the call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67) to the transaction monitor (i.e. the LDAP server) (col 5, lines 50-67), and the transaction monitor (i.e. the LDAP server) (col 5, lines 50-67) relays (col 5, lines 35-67; col 6, lines 35-47) the call (Fig. 2; col 2, lines 16-67) to the directory (i.e. *"In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories."*) (col 2, lines 16-67).

As to claim 22, Ambrosini teaches wherein transaction monitor (i.e. the LDAP server (16), a software entity which is situated in the LDAP server) (Fig. 3; col 5, lines 50-67) relays (col 5, lines 35-67; col 6, lines 35-47) the call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67) to the directory (i.e. "In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories.") (col 2, lines 16-67) through a directory access server (i.e. the LDAP server) (col 5, lines 50-67) that controls access (i.e. *"For example, in order to restrict searches of a directory to entries*

exclusively including access control lists, the search phrase "objectclass=acl" can be specified so that only entries purporting to be access control lists are located.") (col 3, lines 10-17) to the directory (col 2, lines 16-67).

As to claim 23, Ambrosini teaches wherein the rule validator (i.e. plug-in function) (col 5, lines 35-67; col 6, lines 35-47) forwards (i.e. to send the client request to the directory if the validity of the client request is fulfilled) (col 3, lines 25-30; col 6, lines 35-47) the call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67) to the directory (i.e. *"In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories."*) (col 2, lines 16-67).

As to claim 24, Ambrosini teaches wherein the rule validator (i.e. plug-in function) (col 5, lines 35-67; col 6, lines 35-47) forwards (i.e. The LDAP server calls the plug-in function before executing the LDAP operation (e.g. add, delete, modify or search); The plug-in function validates the operation and then sends the validation result to the LDAP server and the server executes the operation on the directory) (col 5, lines 35-67; col 6, lines 35-47) the call (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67) to the directory (i.e. *"In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories."*) (col 2, lines 16-67) through a directory access server (i.e. the LDAP server) (col 5, lines 50-67) that controls access (i.e. *"For example, in order to restrict searches of a directory to entries exclusively including access control lists, the search phrase "objectclass=acl" can be specified so that only entries purporting to be access control lists are located."*) (col 3, lines 10-17) to the directory (col 2, lines 16-67).

As to claim 25, Ambrosini teaches a directory network (i.e. *"The LDAP directory service is based on a client-server model."*) (col 4, lines 15-17), including: one or more client computers (i.e. LDAP clients) (col 4, lines 15-120); a directory (i.e. *"In LDAP, the basic unit of information consists of an*

entry. Entries are stored in a directories.") (col 2, lines 16-67), and an attribute rule enforcer (i.e. plug-in function) (col 5, lines 35-67; col 6, lines 35-47), the attribute rule enforcer (i.e. plug-in function) (col 5, lines 35-67; col 6, lines 35-47) being arranged in the directory network (col 4, lines 15-17) so as to intercept calls (i.e. client's query/ request into the directory database) (Fig. 2; col 2, lines 16-67) from the one or more client computers (i.e. LDAP clients) (col 4, lines 15-120) to the directory (col 2, lines 16-67).

As to claim 26, Ambrosini teaches the directory network (i.e. *"The LDAP directory service is based on a client-server model."*) (col 4, lines 15-17) further including directory access server (i.e. the LDAP server) (col 5, lines 50-67) that controls access (i.e. *"For example, in order to restrict searches of a directory to entries exclusively including access control lists, the search phrase "objectclass=acl" can be specified so that only entries purporting to be access control lists are located."*) (col 3, lines 10-17) to the directory (i.e. *"In LDAP, the basic unit of information consists of an entry. Entries are stored in a directories."*) (col 2, lines 16-67) interposed (i.e. works between the directory and the plug-in function) (col 5, lines 35-67; col 6, lines 35-47) between the attribute rule enforcer (i.e. plug-in function) (col 5, lines 35-67; col 6, lines 35-47) and the directory (col 2, lines 16-67).

Conclusion

5. **THIS ACTION IS MADE FINAL**. See MPEP § 706.07(a). Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not

mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Points of Contact

6. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Apu M. Mofiz whose telephone number is (571) 272-4080. The examiner can normally be reached on Monday – Thursday 8:00 A.M. to 4:30 P.M.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Jeffrey Gaffin can be reached at (571) 272-4146. The fax numbers for the group is (571) 273-8300.

Any inquiry of a general nature or relating to the status of this application should be directed to the Group receptionist whose telephone number is (703) 305-9600.

Apu M. Mofiz
Primary Patent Examiner
Technology Center 2100

August 17,2006